Implementation Details of an Adaptive Flight Controller

Aaron D. Kahn*

U.S. Naval Research Laboratory, Washington, DC, 20375, U.S.A.

The U.S. Naval Research Laboratory has implemented a novel neural-network adaptive flight controller architecture for helicopter-based unmanned aerial vehicles (UAVs) originally presented by Eric Johnson and Suresh Kannan of the Georgia Institute of Technology. Johnson and Kannan utilized on-line adaptation of a neural network to cancel model errors present in an approximate dynamic inversion of the plant. Due to the advantages of risk reduction during controller development and the increased operational flight envelope resulting from the adaptation of the neural-network, this controller was utilized on a number of different rotorcraft at the U.S. Naval Research Laboratory. This paper will describe several interesting implementation details of the controller, as well as modifications made to the basic controller architecture during implementation on different vehicles. Effects of scale, rotor system design, and sample time will be discussed. Simulation and flight test data will be presented showing the results of these modifications.

I. Nomenclature

δ	Control output
x	Vehicle state
$\bar{q}$	Quaternion vector
P, V	Position, Velocity
ω	Angular rate
a, α	Acceleration, Angular Acceleration (Pseudo-controls)
$\hat{a}, \hat{\alpha}$	Approximate pseudo-control
I	Moment of inertia
R	Radius
e	Equivalent flapping hinge offset
P-D	Proportional-Derivative
NED	North, East, Down
A	Linear matrix
g	Gravity vector represented in Earth frame
Ω	Rotor speed (rad/s)
γ	Blade Lock number
τ	Time constant
TPP	Tip-Path-Plane
a_1	Longitudinal rotor TPP flap angle
K_{fb}	Flybar TPP $\rightarrow$ main rotor cyclic mixing gain
K_{cyc}	Main rotor cyclic $\rightarrow$ flybar cyclic mixing gain
M	Mass
b	Span

Subscripts

*Senior Aerospace Engineer, kahn@ccs.nrl.navy.mil, and AIAA Member.

rm	Reference model
h	Hedge
c	Command
pd	Proportional-Derivative compensator
ad	Adaptive
lat, lon, ped	Lateral, Longitudinal, Pedal
col	Collective
ol, il	Outer-loop, Inner-loop
b, fb	Blade, Flybar
des	Desired
$trim$	Trim

Superscripts

B	Body frame
E	Earth frame (North, East, Down)

II. Introduction

Rotary wing vehicles possess many advantages over fixed-wing aircraft for military missions. These advantages include hovering and forward flight, operation out of small spaces, and portability. A limiting factor to the wide-spread use of helicopter unmanned vehicles has been the lack of a flight controller that can reliably handle the wide flight envelope of a helicopter. Recently though, a neural-adaptive flight controller has been developed for helicopter applications.[1,2]

Recent work at the U.S. Naval Research Laboratory has focused on the use of helicopters as practical expendable autonomous vehicles. Two separate air vehicle platforms have been developed, both having different flight dynamics. In order to minimize development time and risk, an adaptive flight controller was chosen for both of these platforms. Due to the small-size of the aircraft, high fidelity flight data was unavailable for system identification techniques to be used in the controller development.[3] A physics-based flight simulator was developed for controller synthesis and testing.[3,4] Due to the possibility of large modeling errors between the simulation model and real vehicle, the flight controller needed to be robust to model errors. In order to satisfy the requirements of risk reduction and robustness, the controller presented by Johnson and Kannan was selected.

This paper will present the implementation details of this particular controller architecture across two vehicle platforms at the U.S. Naval Research Laboratory. The paper will begin with a basic overview of this particular adaptive controller structure. Next, the details of the implementation of specific sections will be addressed. These implementation details will be accompanied by examples taken from flight testing of both research aircraft. Finally, conclusions will be drawn on the practicality of this particular controller when applied across different vehicle platforms.

III. Overview of the Controller

The controller architecture presented by Johnson and Kannan is based on an approximate dynamic inversion of a model of the plant. A neural-network is utilized online to cancel the errors that develop between the approximate model of the plant and the true plant. Due to actuator dynamics, pseudo-control-hedging (PCH) is utilized to eliminate the possibility of mis-adaptation.[5]

For this application, an inner-loop and outer-loop design was used. The inner-loop of the controller regulates attitude of the vehicle via the moment generating controls, δ_{lat},δ_{lon},δ_{ped}. The outer-loop utilizes the rotor thrust generating control, δ_{col}, and the attitude control of the inner-loop for the regulation of position and velocity. An overview of the control architecture can be seen in Figure 1. A more detailed diagram of the inner-loop and outer-loop blocks is illustrated in Figure 2. The reader is pointed to References [1,2,5] for the full development of the controller mathematics. A brief description of the functional elements is given next.

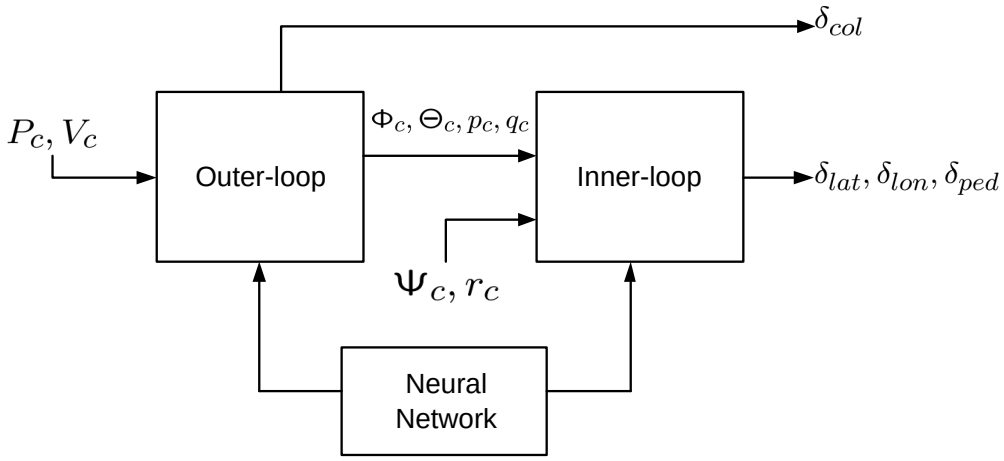

Figure 1. Basic controller architecture.

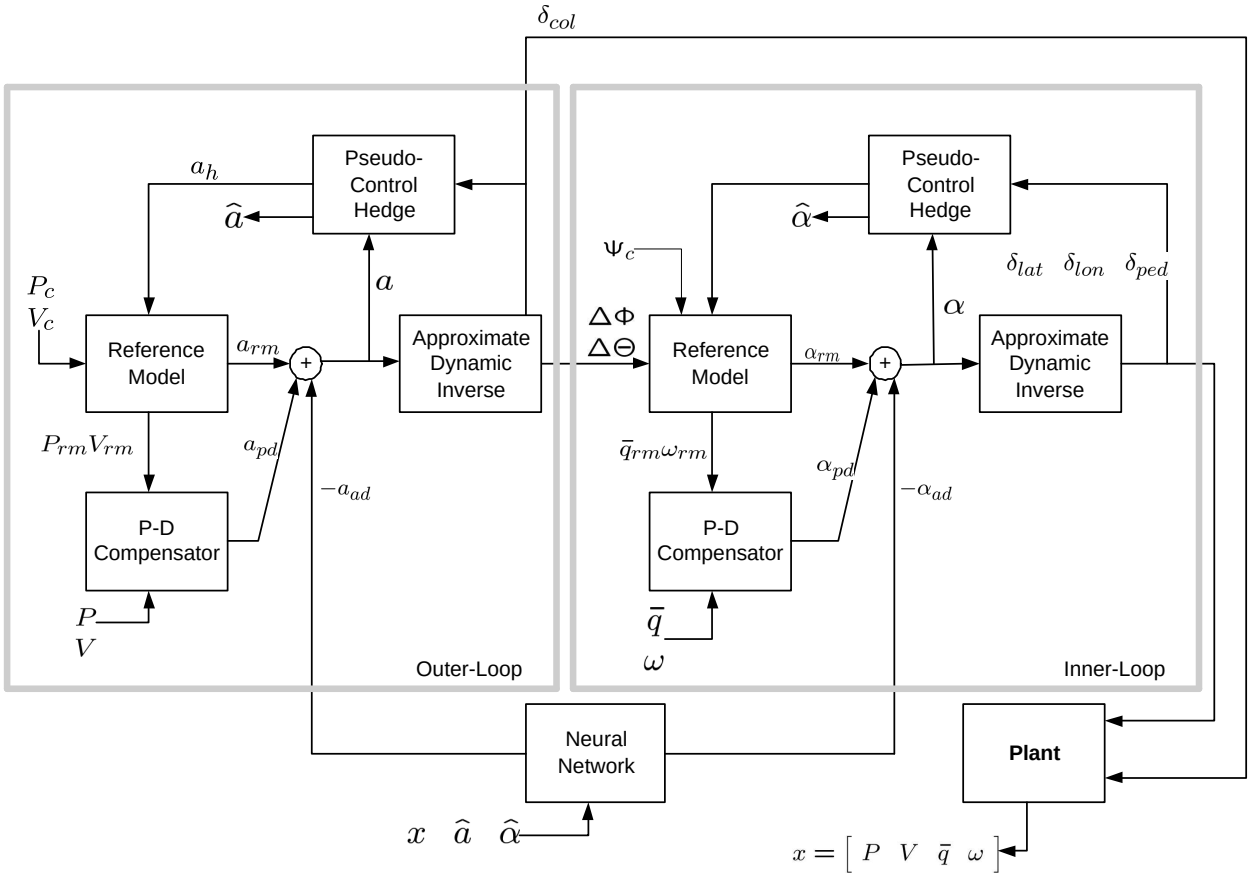

Figure 2. The inner-workings of the adaptive controller.

The controller begins with the outer-loop commands, P_c, V_c. These commands represent the desired position and velocity of the vehicle. The position and velocity commands are prefiltered by the reference model. The internal states of the reference model provide a smooth trajectory for the vehicle to follow. A P-D compensator is utilized to cancel the error between the vehicle state, x, and the reference model states. The total outer-loop pseudo-control, a, is then generated by summing the component pseudo-controls, a_{rm}, a_{pd}, a_{ad}, from the reference model, P-D compensator, and neural network. The pseudo-control from the neural network, a_{ad}, is designed to approximately cancel the residual errors between the approximated dynamic model, used in the inversion block, and the real plant. The total pseudo-control is then utilized in the approximate dynamic inversion block to generate the desired actuator commands. For the outer-loop, these commands are collective angle, δ_{col}, and attitude tilt angles, $\Delta\Phi, \Delta\Theta$. The outer-loop commands of $\Delta\Phi$ and $\Delta\Theta$ plus Ψ_c form the commands of the inner-loop. The inner-loop functionality is the same as the outer-loop, with the resultant outputs being δ_{lat},δ_{lon},δ_{ped} actuator commands.

IV. Implementation of the Adaptive Controller

This controller was implemented in software across two different target computer platforms. The first platform was a PC-104-based avionics system. The second was Cloud Cap Technology's Piccolo II.[6] The Piccolo II is built around the Motorola MPC555 processor. The controller was targeted for a propagation rate of 20 Hz, which is slower than the 50 Hz that Johnson and Kannan used,[1,2] due to the lower computer processing power available. To feed the controller with vehicle data, a 16-state extended Kalman filter was used to estimate the vehicle state. The filter estimated vehicle position, P^E, and velocity, V^E, in the Earth North, East, Down reference frame. Estimated vehicle attitude was represented by quaternions, $\bar{q}$, and bias corrected body angular rate, ω. Three additional states, gravity, north-wind, and east-wind were also estimated. The navigation filter was also propagated at 20 Hz.

A. Vehicles

The two vehicles that this controller was implemented on can be seen in Figures 3 and 4. The Maxi-

Figure 3. Maxi-Joker research vehicle.

Joker aircraft was built around the existing off-the-shelf Maxi-Joker airframe. An unusual aspect of the Maxi-Joker helicopter is that it is electrically powered by a high-efficiency brushless motor, as opposed to

Figure 4. SPIDER (Scientific Payload Insertion Device Electric Rotor) research vehicle.

conventional fuel-powered helicopters. Two Maxi-Joker vehicles were modified to support fully autonomous flight operations. The first, with the PC-104-based avionics system, served as an initial test platform. The second Maxi-Joker supported the Piccolo II avionics system. Both of the Maxi-Joker vehicles were built to serve as surrogate test platforms for the GN&C software before flight on the SPIDER vehicle.

The SPIDER (Scientific Payload Insertion Device Electric Rotor) is a custom designed and built airframe. Like the Maxi-Joker, the SPIDER vehicle is electrically powered. Only the Piccolo II avionics were targeted for the SPIDER airframe. A primary difference between the SPIDER vehicle and the Maxi-Joker, is SPIDER utilizes a variable-speed electric motor for tail-rotor drive, and thus δ_{ped}, as opposed to the conventional collective pitch adjustment of the tail blades used on the Maxi-Joker. This design feature allowed the SPIDER vehicle to efficiently fold into a small package.

The SPIDER and Maxi-Joker vehicle both utilize a semi-rigid rotor system. The blade feathering axle (spindle) is supported by elastomeric dampers (o-rings). A Bell-Hiller flybar is utilized to provide additional stabilization for a manual pilot, via lagged-rate feedback.[3] Table 1 lists some of the important parameters of the SPIDER and Maxi-Joker aircraft.

B. Reference Frames

No mention of the reference frames in which different components were computed was provided in the original description of this controller.[1,2] This reference frame information is important for the correct calculations of both the inner and outer-loop reference models and errors.

The outer-loop commands P_c and V_c are taken in Earth NED frame. This corresponds to the navigation system estimates of position and velocity, P^E and V^E. It was also chosen to propagate the outer-loop reference model states, P_{rm} and V_{rm}, in Earth frame as well. Thus, the error equations for the outer-loop are taken as:

$$e^E_{rm} = P_c - P^E_{rm} \quad (1)$$
$$\dot{e}^E_{rm} = V_c - V^E_{rm} \quad (2)$$
$$e^E = P^E_{rm} - P^E \quad (3)$$
$$\dot{e}^E = V^E_{rm} - V^E \quad (4)$$

The errors calculated from Equations 1-4 are in Earth frame. The pseudo-control, or acceleration, of the outer-loop needs to be in body frame. One might rotate these errors via the full Earth to Body rotation

Component	Maxi-Joker	SPIDER	Units
Main rotor radius	2.85	3.54	ft
Blade lift slope	6.00	5.75	/rad
Blade chord	0.198	0.242	ft
Hinge offset	0.202	0.258	ft
Blade mass	0.01897	0.03842	slug
Flybar radius	1.135	1.208	ft
Flybar paddle lift slope	3.0	3.0	/rad
Flybar paddle chord	0.156	0.197	ft
Flybar paddle span	0.417	0.346	ft
Flybar inertia	0.001219	0.0043	slug-ft^2
Flybar TPP $\rightarrow$ cyclic	4.225	4.03	rad/rad
cyclic $\rightarrow$ flybar cyclic	0.716	0.603	rad/rad
Rotor speed	1000	920	RPM
Vehicle mass	0.3727	1.118	slug
I_{xx}	0.0845	0.9313	slug-ft^2
I_{yy}	0.1901	2.937	slug-ft^2
I_{zz}	0.1065	0.9700	slug-ft^2

Table 1. Parameters of Maxi-Joker and SPIDER vehicles.

matrix in Equation 5

$$C^{BE} = \begin{bmatrix} 1-2(q_2^2+q_3^2) & 2(q_1q_2+q_0q_3) & 2(q_1q_3-q_0q_2) \\ 2(q_1q_2-q_0q_3) & 1-2(q_1^2+q_3^2) & 2(q_2q_3+q_0q_1) \\ 2(q_1q_3+q_0q_2) & 2(q_2q_3-q_0q_1) & 1-2(q_1^2+q_2^2) \end{bmatrix} \tag{5}$$

If this is done, an error in altitude will appear as a function of horizontal error and attitude tilt. This effect is graphically shown in Figure 5. This induced error would result in the vehicle either climbing or descending.

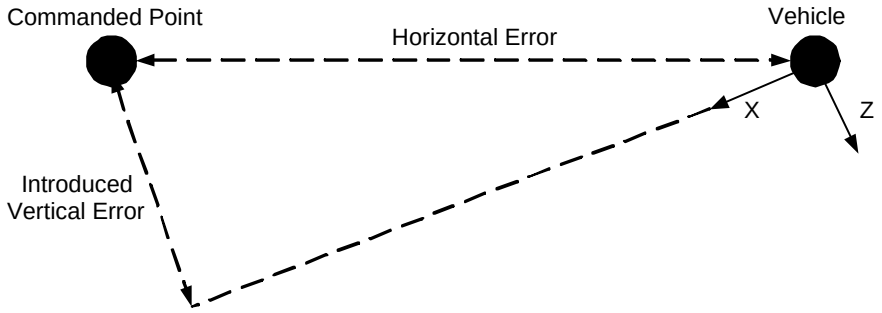

Figure 5. Altitude error introduced by full rotation matrix.

This effect can be seen in Figure 6 which is from a position step command of the Maxi-Joker while in flight at $t = 667sec$. To prevent this from occurring, only a partial rotation matrix is utilized. This matrix is shown in Equation 7.

$$\Psi \quad = \quad \tan\left(\frac{2(q_1q_2+q_0q_3)}{1-2(q_2^2+q_3^2)}\right)^{-1} \tag{6}$$

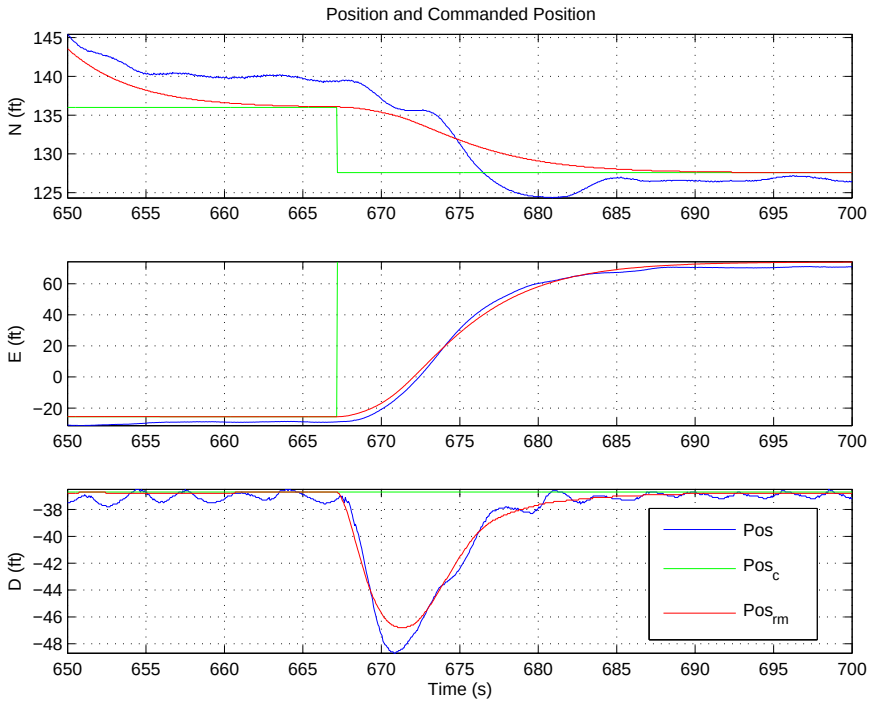

Figure 6. Altitude jump caused by induced altitude error.

$$Z^{BE} = \begin{bmatrix} \cos(\Psi) & \sin(\Psi) & 0 \\ -\sin(\Psi) & \cos(\Psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{7}$$

The errors are then finally rotated to body frame as:

$$e^B_{rm} = Z^{BE} e^E_{rm} \tag{8}$$
$$\dot{e}^B_{rm} = Z^{BE} \dot{e}^E_{rm} \tag{9}$$
$$e^B = Z^{BE} e^E \tag{10}$$
$$\dot{e}^B = Z^{BE} \dot{e}^E \tag{11}$$

With the errors now rotated correctly into the body frame, the rest of the outer-loop can be calculated as shown in References [1, 2].

As stated earlier the reference model states are propagated in Earth frame, yet the reference model acceleration is in body frame. The full rotation matrix is utilized to convert from body frame back to Earth frame. The dynamic equations for the outer-loop reference model are shown in Equations 12-13.

$$\dot{V}^E_{rm} = C^{BE^T} a_{rm} \tag{12}$$
$$\dot{P}^E_{rm} = V^E_{rm} \tag{13}$$

Unlike the outer-loop, the inner-loop operates only in the body frame. The commands to the inner-loop reference model are desired pitch and roll angles from the outer-loop, and the commanded heading. These Euler-angles are converted to a quaternion, $\bar{q}_{des}$, for the final command. Because the inner-loop attitude command is represented by $\bar{q}_{des}$, the total rotation caused by these three Euler-angles is the utilized. This has the effect of rotating the attitude commands, $\Delta\Phi$ and $\Delta\Theta$ by the commanded yaw angle Ψ_c. This effect is illustrated in Figure 7. As can be seen, the commanded pitch and roll attitude angles need to be rotated such that this effect is avoided. Using the reference model quaternion, $\bar{q}_{rm}$, as the argument for Equation 6, one can compute the current reference model heading, Ψ_{rm}. The error between the reference model and

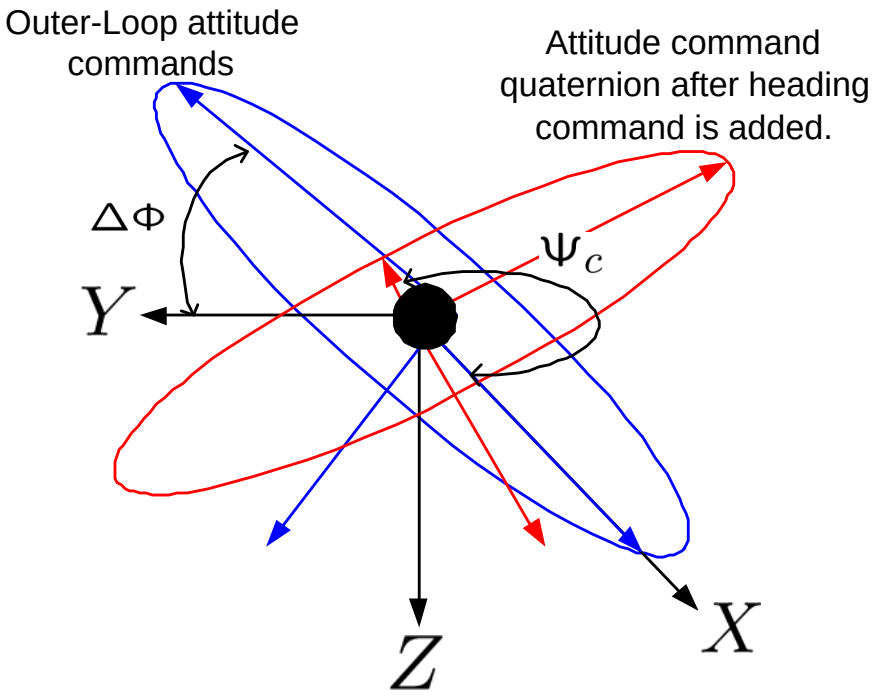

Figure 7. Effect of included heading command in attitude quaternion.

commanded headings is:

$$\Delta\Psi_{rm} = tohrev(\Psi_{rm} - \Psi_c) \tag{14}$$

where $tohrev(\cdot)$ modulates a number between $-\pi$ and π. The outer-loop attitude commands are then rotated using Equation 15.

$$\begin{bmatrix} \Phi_c \\ \Theta_c \end{bmatrix} = \begin{bmatrix} \cos(\Delta\Psi_{rm}) & -\sin(\Delta\Psi_{rm}) \\ \sin(\Delta\Psi_{rm}) & \cos(\Delta\Psi_{rm}) \end{bmatrix} \begin{bmatrix} \Delta\Phi \\ \Delta\Theta \end{bmatrix} \tag{15}$$

The final inner-loop attitude quaternion command is then generated as:

$$\bar{q}_{des} = Q(\Phi_c, \Theta_c, \Psi_c) \tag{16}$$

where $Q(\cdot)$ is the function to convert from Euler-angles to quaternion.[7] This rotation of the attitude command angles allows the vehicle to operate correctly during large displacement steps in heading, or during prolonged durations of yaw actuator saturation. Such a situation occurred on the SPIDER vehicle, and is shown in Figure 8. Because of the design of the SPIDER vehicle's tail-rotor, it cannot produce 0 and negative thrust. It relies on the main rotor torque, which is proportional to main rotor collective, to allow the vehicle to rotate to the right. As main rotor collective is reduced, the resulting torque on the body is reduced. If the collective is reduced beyond a point, this torque is reduced and the vehicle is unable to arrest the resulting left pirouette until the collective is returned to a higher value.

C. Update Rate

The controller design was done using continuous-time analysis. When implemented on a digital computer the rate at which the controller is propagated must be sufficiently faster than the dynamics which it is controlling. For the case of the original literature, the controller was propagated at a rate of 50 Hz.[1,2] Due to the limited computing resources on both of these aircraft, this implementation only propagated the controller at 20 Hz. This reduction in update rate resulted in an interesting problem of the pseudo-control hedging algorithm.

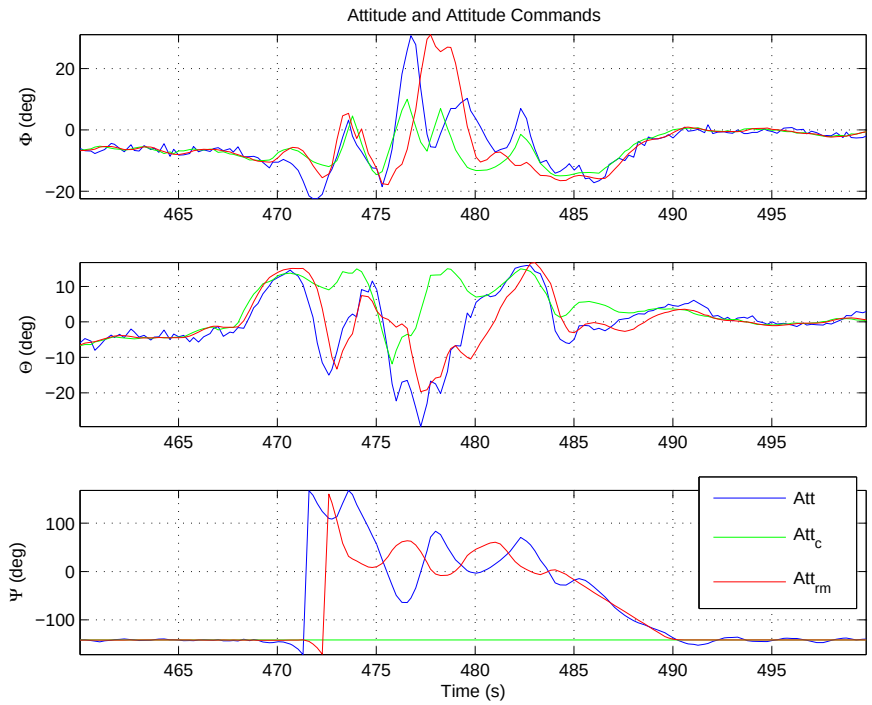

Figure 8. SPIDER attitude during tail-rotor saturation.

As presented, the pseudo-control hedge[5] works by first computing an estimate of the achievable total pseudo-control, $\hat{\nu}$ by either an estimate or measurement of the current actuator value, δ. By taking the difference between the desired total pseudo-control, ν, and estimated achievable total pseudo-control, $\hat{\nu}$, a pseudo-control hedge, ν_h, is then generated. This hedge is then used to modify the reference model such that the total desired pseudo-control becomes equal to the achievable pseudo-control. This process is done to prevent the neural-network from mis-adapting to events such as actuator saturation. If the approximate dynamic model is given by $f(\cdot)$, and the actuator model is $g(\cdot)$, then the pseudo-control hedge calculations follow as:

$$\delta = f^{-1}(\nu, x) \tag{17}$$

$$\hat{\delta} = g(\delta, x) \tag{18}$$

$$\hat{\nu} = f(\hat{\delta}, x) \tag{19}$$

$$\nu_h = \nu - \hat{\nu} \tag{20}$$

$$\ddot{x}_{rm} = \nu_{rm} - \nu_h \tag{21}$$

The pseudo-control hedging equations rely on the update rate of the control system being sufficiently fast that it can be thought of as continuous. This is particularly true because the hedge can generate a discontinuity in the acceleration of the reference model. The successful propagation of the reference model in time when implemented on a digital computer is only guaranteed if the instantaneous reference model acceleration stays below a value $\bar{\ddot{x}}_{rm}$. If this bound is exceeded, the reference model integration will diverge. This limit is present due to the discretization of the dynamics, and becomes more apparent as the time step becomes larger, or the update rate slows. Under normal operation, this problem is not visible as the reference model dynamics are smooth, and designed such that they can be propagated successfully at the update rate of the controller. If, though, the difference between ν and $\hat{\nu}$ becomes large in one time step, such as in the event of actuator saturation, the reference model acceleration can be pushed beyond $\bar{\ddot{x}}_{rm}$.

To prevent this from occurring, the original pseudo-control hedge equations were augmented with the

pseudo-control hedge acceleration limiter. This augmentation can be seen in Equation 22.

$$\ddot{x}_{rm} = \begin{cases} \bar{\ddot{x}}_{rm} & : \quad \nu_{rm} - \nu_h > \bar{\ddot{x}}_{rm} \\ \nu_{rm} - \nu_h & : \quad -\bar{\ddot{x}}_{rm} \leq \nu_{rm} - \nu_h \leq \bar{\ddot{x}}_{rm} \\ -\bar{\ddot{x}}_{rm} & : \quad \nu_{rm} - \nu_h < -\bar{\ddot{x}}_{rm} \end{cases} \tag{22}$$

This problem was discovered as a result of the loss of one Maxi-Joker vehicle during a test flight. During the flight, the δ_{ped}actuator saturated very rapidly, due to the high slew-rate of that particular actuator. This, via the pseudo-control hedge, caused a large jump in the reference model acceleration. This resulted in the propagation of the reference model diverging, and thus the failure of the controller. This failure of the reference model can be seen in Figure 9.

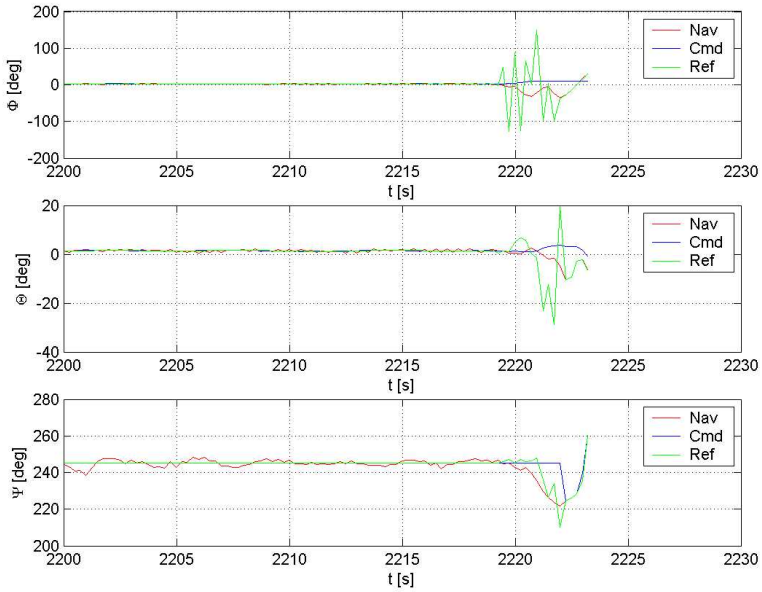

Figure 9. Inner-loop reference model diverging due to PCH hedge.

It was also found that the parameters for the neural-network were dependent upon the update rate of the controller. This was especially true for the bias weight b_w and κ.[1,2] The formulation of the neural-network can be found in References [1, 2]. Importantly, it was discovered that if κ was too small, the propagation of the network would diverge. This value can be thought of as a damping term. It was found empirically that κ should be approximately equal to the time step of the controller. In this particular implementation, κ was chosen to be 0.05. The bias value, b_w, which scales the input to the adaptation law, was found to prevent jitters in the adaptation. The larger b_w was made, the less active the network became. If b_w was made too small, the adaption output became erratic. The network parameters that were implemented on both the Maxi-Joker and SPIDER vehicles are listed in Table 2. As can be seen in Table 2 the robustifing signal as described in References [1, 2] is not used. During simulation and flight test it was found not to be needed.

D. Approximate Dynamic Model

A key benefit stated of this particular adaptive controller is the ability to use a poor model of the plant as the dynamic inverse. The approximate model used for the inner-loop is Equation 23, and the outer-loop

Parameter	Value
b_v	0.5
b_w	3.5
κ	0.05
Γ_V	10.0
Γ_W	1.0
K_r	0

Table 2. Neural-network parameters used on both Maxi-Joker and SPIDER vehicles.

model used is Equation 24.[1]

$$\alpha = A\omega + B\left(\underbrace{\begin{bmatrix}\delta_{lat}\\ \delta_{lon}\\ \delta_{ped}\end{bmatrix}}_{des} - \underbrace{\begin{bmatrix}\delta_{lat}\\ \delta_{lon}\\ \delta_{ped}\end{bmatrix}}_{trim}\right) \tag{23}$$

$$a = \begin{bmatrix}0\\ 0\\ Z_{\delta_{col}}\end{bmatrix}(\delta_{col_{des}} - \delta_{col_{trim}}) + C^{BE}g \tag{24}$$

The value $Z_{\delta_{col}}$ is the estimated control power of the main rotor collective. The matrix A is the linearized estimated rate damping values and the matrix B is the estimated inner-loop control power and allocation matrix.

During the early implementation of the controller, the inner-loop model 23 was simplified by setting A to 0. It was found during flight testing that this model was not sufficient. The reason for this can be seen in a linear simulation of the system. In this simulation, only the inner-loop pitch dynamics were examined, and the reference model states were assumed to be steady. This simulation also neglected the effect of the neural-network, as only the stability of the basic dynamic inversion was evaluated. The P-D compensator gains were calculated based on the equations presented in the literature[1,2] and shown in Equations 25 and 26.

$$K_p = \omega_{il}^2 + 4\zeta_{ol}\omega_{ol}\zeta_{il}\omega_{il} + \omega_{ol}^2 \tag{25}$$

$$K_d = 2\zeta_{il}\omega_{il} + 2\zeta_{ol}\omega_{ol} \tag{26}$$

This simulation will utilize the SPIDER parameters from Table 1 for the vehicle model. The linear model used is based upon the rotor flapping and flybar teetering dynamics.[3,4,8] The model can be seen in Equations 27-36.

$$\Omega = \frac{2RPM\pi}{60} \tag{27}$$

$$I_b = \frac{M_b}{R_b}\frac{(R_b - e)^3}{3} \tag{28}$$

$$\gamma_{fb} = \frac{\rho a_{fb} c_{fb}[R_{fb}^4 - (R_{fb} - b_{fb})^4]}{I_{fb}} \tag{29}$$

$$\tau_{fb} = \frac{16}{\Omega\gamma_{fb}} \tag{30}$$

$$\gamma_b = \frac{\rho a_b c_b R_b^4}{I_b} \tag{31}$$

$$\tau_b = \frac{16}{\Omega\gamma_b} \tag{32}$$

$$\frac{dM}{da_1} = 1.5 I_b \frac{e}{R_b} \Omega^2 \tag{33}$$

$$\frac{dL}{db_1} = \frac{dM}{da_1} \tag{34}$$

$$X = \begin{bmatrix} a_{1_{fb}} \\ a_1 \\ q \\ \Theta \end{bmatrix} \tag{35}$$

$$\dot{X} = \begin{bmatrix} \frac{-1}{\tau_{fb}} & 0 & -1 & 0 \\ \frac{K_{fb}}{\tau_b} & \frac{-1}{\tau_b} & -1 & 0 \\ 0 & \frac{\frac{dM}{da_1}}{I_{yy}} & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} X + \begin{bmatrix} \frac{K_{cyc}}{\tau_{fb}} \\ \frac{1}{\tau_b} \\ 0 \\ 0 \end{bmatrix} \delta_{lon} \tag{36}$$

Figure 10 illustrates the time-response of pitch attitude to a step pitch attitude command using three different dynamic models. Figure 11 shows the block diagram for this simulation. An inner-loop natural frequency,

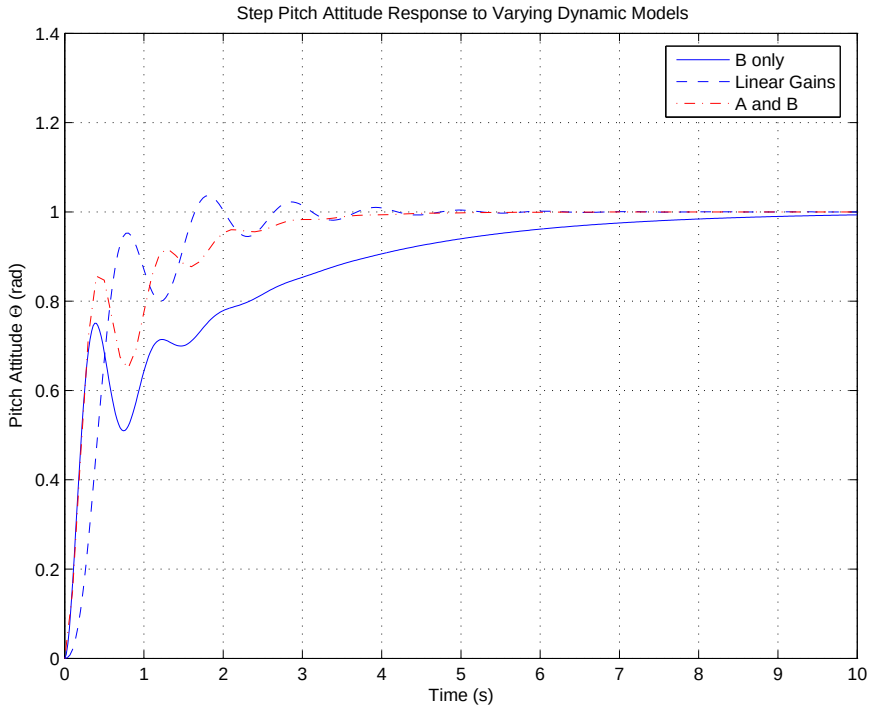

Figure 10. Pitch attitude step response with various P-D compensation.

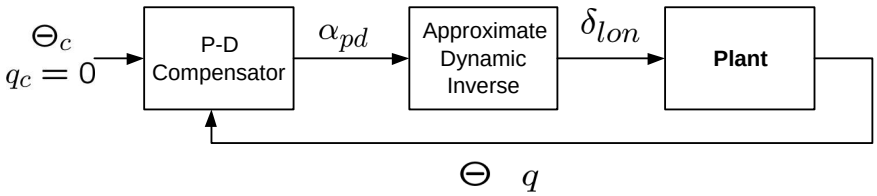

Figure 11. Linear simulation block diagram.

$\omega_{n_{il}}$, of 1.0, and outer-loop natural frequency, $\omega_{n_{ol}}$, of 0.6 were used. Both the inner and outer-loop damping

ratios were set to unity. These values were used in Equations 25 and 26 for the calculation of the P-D compensator gains. The first trace shows an unsatisfactory rise time when using just the B matrix alone. This model/P-D compensator is shown in Equations 37 and 38. A value of 44.3 $(rad/s^2)/rad$ was used for B. The primary reason for this slow response is the damping effect of the flybar in the rotor system.

$$\delta_{lon} = \frac{\alpha_{pd}}{B} \tag{37}$$

$$\alpha_{pd} = K_p(\Theta_c - \Theta) + K_d(q_c - q) \tag{38}$$

To improve this slow tracking performance, the gains K_p and K_d were explicitly calculated. The simple B-only model as shown in Equation 37 was still used. To allow the explicit values for K_p and K_d to be used, gain multipliers were computed. Equations 39 and 40 show the calculation of these given the desired proportional and derivative gains, $K_{p_{lin}}$ and $K_{d_{lin}}$. Values of 0.136 and 0.002 for $K_{p_{lin}}$ and $K_{d_{lin}}$ respectively were used. These values were obtained from flight testing of the SPIDER vehicle.

$$iK_p = \frac{K_{p_{lin}}B}{K_p} \tag{39}$$

$$iK_d = \frac{K_{d_{lin}}B}{K_d} \tag{40}$$

$$\alpha_{pd} = iK_pK_p(\Theta_c - \Theta) + iK_dK_d(q_c - q) \tag{41}$$

$$\tag{42}$$

The result of this modification can be seen in Figure 10. A problem arises when this modification is made to the P-D compensator. In the formulation of the neural-network adaptation law, the gains in the P-D compensator are used.[1,2] By using these multipliers a difference between the adaptation law and the P-D compensator is created. This results in the neural-network being unable to cancel the model error correctly. This result was found during high-speed flight testing. The neural-network was unable to adapt to the changing vehicle dynamics as a function of flight speed.

The correct modification is to make the dynamic model slightly more complex with the addition of A. This value allows the designer to effectively modify the ratio of K_p to K_d without effecting the formulation of the neural-network adaptation law. The new augmented dynamic inverse is can be seen in Equation 43.

$$\delta_{lon} = \frac{\alpha_{pd} - Aq}{B} \tag{43}$$

If one substitutes the P-D compensator, Equation 38, for α_{pd} the result is:

$$\delta_{lon} = \frac{K_p(\Theta_c - \Theta) + K_dq_c - q(K_d + A)}{B} \tag{44}$$

It can be seen from Equation 44 that a negative value for A will have the effect of reducing the feedback effect of K_d. In this simulation, the value q_c was set to 0. A value of -2.7 $(rad/s^2)/(rad/s)$ was used for A and 52 $(rad/s^2)/rad$ was used for B. These values were obtained from flight testing. The resulting step response can be seen in Figure 10.

From this simulation, it can be seen that the proper choice of dynamic model is critical. During flight testing, it was found useful to disable the neural-network and rely solely on the P-D compensator control. This allowed for the observing of the performance of the baseline dynamic inverting controller.

With this same linear simulation, an initial selection of the inner-loop dynamic model parameters can be solved for. This method is designed to solve for the two parameters, A and B, using the simplified pitch (or roll) dynamics from Equation 36. The response of the pitch-rate (or roll rate) is used instead of pitch (or roll) attitude. The reason for this can be seen in Equation 45.

$$\dot{q} = Aq + B\delta_{lon} \tag{45}$$

The original 3^{rd}-order model, with respect to rate, is reduced to first-order. The responses of the pitch models for both the SPIDER and Maxi-Joker to a step input can be seen in Figures 12 and 13 respectively. The data listed in Table 1 was used in the creation of the pitch dynamic models. Table 3 lists the estimated and actual flight parameters for the pitch axis of both the SPIDER and Maxi-Joker. It can be seen in Table 3 that the estimated rate damping was consistently about 1.6-times larger than the parameter found from flight testing. The control power from the model closely matches that found from flight testing.

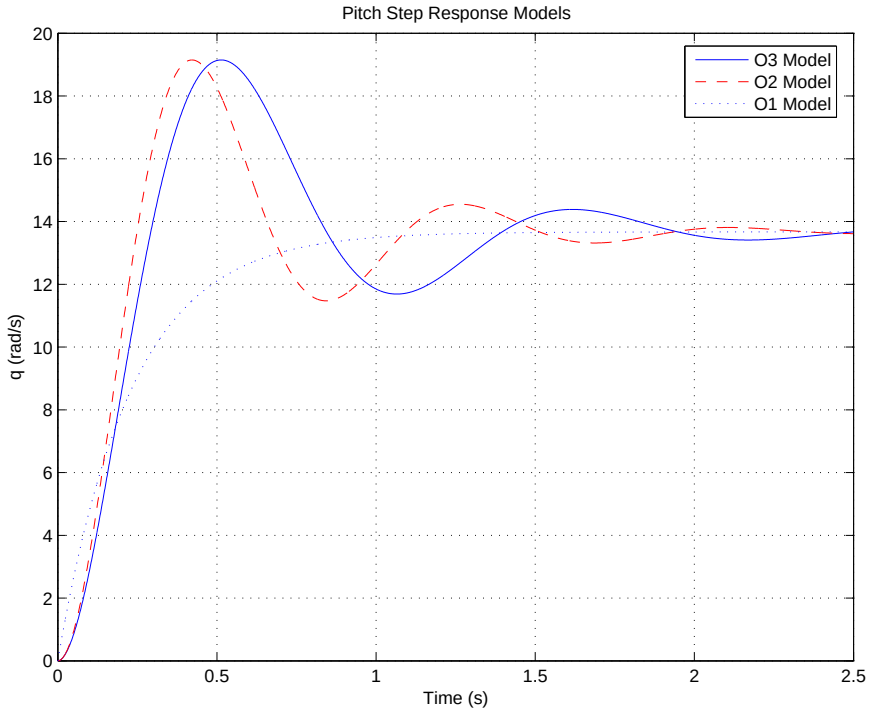

Figure 12. Reduced order pitch model of SPIDER.

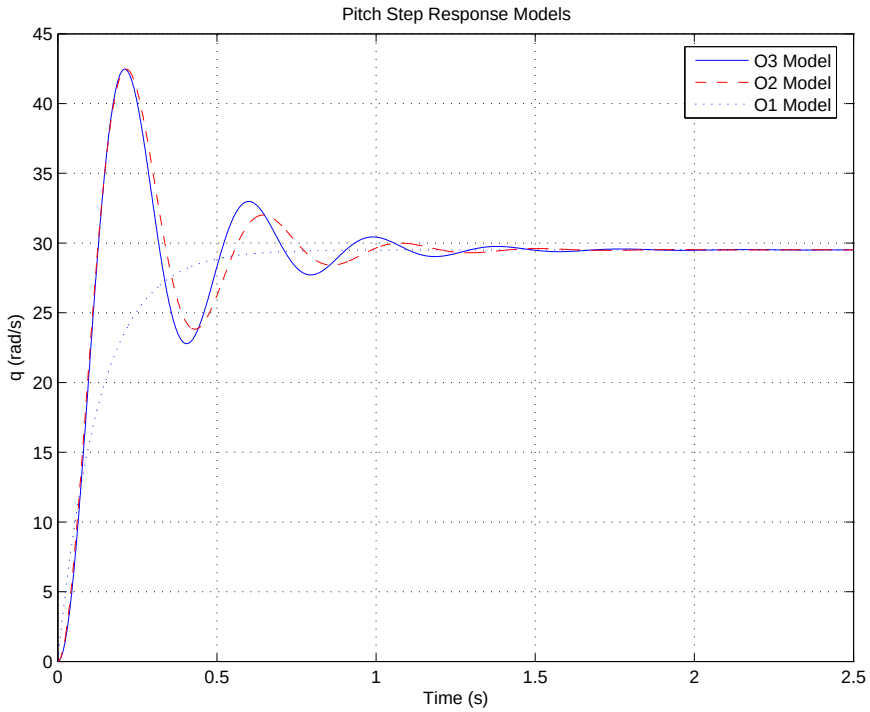

Figure 13. Reduced order pitch model of Maxi-Joker.

Vehicle	A Estimated	A Flight	B Estimated	B Flight
Maxi-Joker	-7.648	-4.5	225.7	207
SPIDER	-4.336	-2.7	59.27	52

Table 3. Flight and estimated pitch parameters for both SPIDER and Maxi-Joker.

V. Conclusion

In order to reduce both development risk and time, an adaptive controller was chosen for use on two vehicles at the U.S. Naval Research Laboratory. The controller implemented was based on an inner and outer-loop architecture originally presented by Johnson and Kannan.[1,2] By using a neural-network, model errors present in the approximate dynamic inversion are canceled online. This design reduced the dependency on having an accurate mathematical model of the vehicle during development.

During the implementation of the controller, three key points were addressed: reference frame selection, rate of controller propagation, and selection of the approximate dynamic model. Two of these three areas caused some difficulty in successful implementation of the controller. First, the rate of controller propagation caused issues in the stability of the propagation of the reference model during hedging events. An acceleration limiter was invented to prevent the pseudo-control hedge from causing a divergence of the reference model. The choice of neural-network parameters was also found to be dependent on the update rate of the controller. Next, several iterations of inner-loop approximate dynamic inverse models were investigated. It was found that the model needed a minimum level of complexity to accurately capture the effect of the rotor system flybar damping, while not violating the derivation of the neural-network adaptation law.

Once these implementation details were solved, the controller performance on these two research vehicles has been acceptable. When the achieved performance of this controller/vehicle combination is compared to similar vehicles implementing basic linear control, such as PID, the usable flight envelope is much larger. Currently, both research vehicles have flown in excess of 44 miles-per-hour airspeed with only a controller designed and tuned around the hover flight condition. To date, three SPIDER airframes have been flown with this controller, and no controller parameter adjustments were required across these platforms. In conclusion, given the level of achievable performance for the complexity, this particular controller implementation is quite practical.

VI. Acknowledgments

The author wishes to thank Suresh Kannan and Dr. Eric Johnson in answering questions during the development process of this work. The author also thanks Ross Hoag, Bill Vaglienti, and Marius Niculescu at Cloud Cap Technology for their support. Thanks to Steven Tayman, Chris Bovais, and others at the U.S. Naval Research Laboratory that made these flight tests possible.

References

[1]Johnson, E. N. and Kannan, S., "Adaptive Flight Control for an Autonomous Unmanned Helicopter," No. AIAA-2002-4439, Monterey, CA, aug 2002, http://controls.ae.gatech.edu/papers/kannan_gnc_02.pdf.

[2]Kannan, S. and Johnson, E. N., "Adaptive Trajectory Based Flight Control for Autonomous Helicopters," Irvine, CA, oct 2002, http://controls.ae.gatech.edu/papers/kannan_dasc_02.pdf.

[3]Mettler, B., Tischler, M., and Kanade, T., "System Identification of a Model-Scale Helicopter," Tech. Rep. CMU-RI-TR-00-03, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, January 2000, http://www.ri.cmu.edu/pubs/pub_3328.html.

[4]Heffley, R. K. and Mnich, M. A., "Minimum-Complexity Helicopter Simulation Math Model," Tech. Rep. CR 177476, USAAVSCOM TR 87-A-7, NASA, California, apr 1988.

[5]Johnson, E. N., *Limited Authority Adaptive Flight Control*, Ph.D. thesis, Georgia Institute of Technology, School of Aerospace Engineering, Atlanta, GA 30332, dec 2000, http://www.ae.gatech.edu/∼ejohnson/Thesis-repro.pdf.

[6]Technology, C. C., "Cloud Cap Technology: Downloads," Tech. rep., 2005, http://www.cloudcaptech.com/downloads.htm.

[7]Stevens, B. L. and Lewis, F. L., *Aircraft Control and Simulation*, John Wiley and Sons, New York, 1992.

[8]Prouty, R. W., *Helicopter Performance, Stability, and Control*, Krieger Publishing Company, Malabar, Florida, 1995.